PATENT
Docket No. **SYB/0102.01**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Vadiraja P. Bhatt et al.	Examiner: Ahmed, Hamdy S
Serial No.: 10/710,381	Art Unit: 2188
Filed: July 6, 2004	APPEAL BRIEF
For: Database System Providing Methodology for Extended Memory Support	

Mail Stop Appeal
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

BRIEF ON BEHALF OF VADIRAJA P. BHATT, ET AL

This is an appeal from the Final Rejection mailed October 31, 2007, in which currently-pending claims 1-45 stand finally rejected. Appellant filed a Notice of Appeal on February 4, 2008. This brief is submitted electronically in support of Appellant's appeal.

TABLE OF CONTENTS

1. REAL PARTY IN INTEREST

The real party in interest is assignee Sybase, Inc. located at One Sybase Drive, Dublin, CA 94568.

2. RELATED APPEALS AND INTERFERENCES

There are no appeals or interferences known to Appellant, the Appellant's legal representative, or assignee which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

3. STATUS OF CLAIMS

The status of all claims in the proceeding is as follows:

Rejected: Claims 1-45

Allowed or Confirmed: None

Withdrawn: None

Objected to: None

Canceled: None

Identification of claims that are being appealed: 1-45

An appendix setting forth the claims involved in the appeal is included as Section 8 of this brief.

4. STATUS OF AMENDMENTS

One Amendment and one Response and Request for Reconsideration have been filed in this case. Appellant filed an Amendment on February 9, 2007 in response to a non-final Office Action dated November 9, 2006. In the Amendment filed by Appellant on February 9, 2007, the claims were amended in a manner which Appellant believes clearly distinguishes Appellant's claimed invention from the art of record, for overcoming the art rejections. In response to a second non-final Examiner Office Action dated May 3, 2007 Appellant filed a Response and Request for Reconsideration on

August 27, 2007. In response to the Examiner Office Action dated October 31, 2007 (hereinafter "Final Rejection") finally rejecting Appellant's claims, Appellant filed a Notice of Appeal. Appellant has chosen to forego filing an Amendment After Final as it is believed that further amendments to the claims are not warranted in view of the art. Accordingly, no amendments have been entered in this case after the date of the Final Rejection.

5. SUMMARY OF CLAIMED SUBJECT MATTER

As to Appellant's **First Ground** for appeal, Appellant asserts that the art rejection under 35 U.S.C. Section 103(a) relying on the combination of U.S. Published Application 20040103251 A1) of Alsup (hereinafter "Alsup") and U.S. Patent 7,124,249 to Darcy (hereinafter "Darcy") fails to teach or suggest all of the claim limitations of Appellant's rejected claims 1-3, 5-26 and 28-45, where the claimed invention is set forth in the embodiment in **independent claim 1:** A method for extended memory support in a database system (Appellant's specification paragraph [0019], paragraphs [0066]-[0067], paragraph [0068] (embodied in database system and provides extended memory support); see generally paragraphs [0188]-[0189]; also see generally Fig. 3 (database system) and Fig. 5) and having a primary cache for storing database pages (Appellant's specification paragraph [0019], paragraphs [0065]-[0066], paragraph [0104]; Fig. 5 at 520; Fig. 6A at 620; Fig. 6B at 620a) the method comprising: using a memory mapped file, creating a secondary cache in system memory available to the database system (Appellant's specification paragraph [0019], paragraph [0066], paragraph [0067] (memory mapped file feature), paragraph [0088]; Fig. 5 at 530; Fig. 6A at 630; Fig. 6B at 630a), mapping a virtual address range to at least a portion of the secondary cache (Appellant's specification paragraph [0019], paragraphs [0067] (maps virtual address range to portion of shmfs file), paragraph [0089], paragraph [0092]; see generally paragraphs [0167]-[0169]; see also, Fig. 5 and Fig. 11), when the primary cache is full, replacing database pages from the primary cache using the secondary cache (Appellant's specification paragraph [0019], paragraph [0066] (pages from primary cache spill over to secondary cache), paragraph [0067] (memory mapped file space is used as a secondary cache from which the system replaces the pages from the primary cache), paragraphs [0104]-[0108], paragraph [0112]; Fig. 5 and Fig. 6B), in response to a request for a particular database page, searching for the database particular page in the secondary cache if the particular database page is not found in the primary cache (Appellant's specification paragraph [0019], paragraph [0066], paragraph [0122] (secondary cache searched when primary cache does not contain desired page); Fig. 8; Fig. 9A at 901, 902; Fig. 13 at 1303), if the particular database page is found in the secondary cache, determining a virtual address in the secondary cache where the particular database page resides based on the mapping (Appellant's specification paragraph [0019], paragraph [0067] (if page in secondary cache, system maps virtual address window to where page resides), paragraph [0089], paragraph [0092], paragraph [0105], paragraph [0186]; Fig. 5, Fig. 8, Fig. 13 at 1303, 1304), and swapping the particular database page found in the secondary cache with a database page in the primary cache, so as to replace a database page in the primary cache with the particular database page from the secondary cache (Appellant's specification paragraph [0019], paragraph [0067], paragraphs [0122]-[0125], paragraphs [0126]-[0128], paragraph [0186]; Fig. 5, Fig. 8; Figs. 9A-B at 903-908; Fig. 13 at 1304).

For Appellant's argument under the **First Ground** for appeal, Appellant additionally argues that the art rejection under 35 U.S.C. Section 103(a) relying on the combination of Alsup (above) and Darcy (above) fails to teach or suggest all of the claim limitations of Appellant's rejected claims 1-3, 5-26 and 28-45, where the claimed invention is set forth in the embodiment in **independent claim 24:** A database system providing extended memory support (Appellant's specification paragraph [0020], paragraphs [0066]-[0067], paragraph [0068] (embodied in database system and provides extended memory support); see generally paragraphs [0188]-[0189]; also see generally Fig. 3 (database system) and Fig. 5), the system comprising: a primary cache for maintaining data pages used by the database system in addressable memory available to the database system (Appellant's specification paragraph [0020], paragraphs [0065]-[0066], paragraph [0104]; Fig. 5 at 520; Fig. 6A at 620; Fig. 6B at 620a), a secondary cache, created in system memory using a memory mapped file, for maintaining data pages replaced from the primary cache in extended memory available to the database system (Appellant's specification paragraph [0020], paragraph [0066], paragraph [0067] (memory mapped file feature), paragraph [0088]; Fig. 5 at 530; Fig. 6A at 630; Fig. 6B at 630a), a search module for receiving a request from a user for a particular data page and determining whether the particular data page is in secondary cache if the particular data page is not in the primary cache (Appellant's specification paragraph [0020], paragraph [0066], paragraph [0122] (secondary cache searched when primary cache does not contain desired page), paragraph [0186]; Fig. 8; Fig. 9A at 901, 902; Fig. 13 at 1303), and a module for replacing a data page in the primary cache with the particular data page from the secondary cache if the particular data page is found in the secondary cache (Appellant's specification paragraph [0020], paragraph [0067], paragraphs [0122]-[0125], paragraphs [0126]-[0128], paragraph [0186]; Fig. 5, Fig. 8; Figs. 9A-B at 903-908; Fig. 13 at 1304).

For Appellant's argument under the **First Ground** for appeal, Appellant additionally argues based on **dependent claims 2 and 25** which include limitations of creating the secondary cache using a shared memory file system (Appellant's specification paragraphs [0067]-[0068], paragraph [0083], paragraph [0088], paragraph [0104]; Fig. 5). For Appellant's argument under the **First Ground** for appeal, Appellant additionally argues based on **dependent claims 3 and 26** which include limitations of a shared memory file system is available as part of an operating system on a computer platform on which the database system is running (Appellant's specification paragraph [0037], paragraphs [0067]-[0068], paragraph [0070]). For Appellant's argument under the **First Ground** for appeal, Appellant also argues based on **dependent claim 18** which includes limitations of determining database pages to be maintained in the secondary cache based on workload of the database system (Appellant's specification paragraph [0093], paragraph [0102]).

As to Appellant's **Second Ground** for appeal, Appellant asserts that the art rejection under **Section 103(a)** relying on the combination of Darcy (above) and US Published Application No: 20030162544 A1 of Austin et al. (hereinafter "Austin") fails to teach or suggest all of the claim limitations of Appellant's rejected claims 4 and 27, where the claimed invention is set forth in the embodiment in **independent claims 1 and 24** (the mapping of which is shown above under Appellant's **First Ground** for appeal, and which hereby is incorporated by reference). For Appellant's argument under the

First Ground for appeal, Appellant additionally argues based on **dependent claims 4 and 27** which include limitations of utilizing the Linux operating system (Appellant's specification paragraph [0065], paragraphs [0067]-[0068], paragraph [0070]).

6. GROUNDS OF REJECTION TO BE REVIEWED

The grounds for appeal are:

(1st) Whether claims 1-3, 5-26 and 28-45 are unpatentable under 35 U.S.C. Section 103(a) over U.S. Published Application 20040103251 A1 of Alsup (hereinafter "Alsup") in view of U.S. Patent 7,124,249 to Darcy (hereinafter "Darcy"); and

(2nd) Whether claims 4 and 27 are unpatentable under 35 U.S.C. 103(a) as being unpatentable over Darcy (above) in view of US Published Application No: 20030162544 A1 of Austin et al. (hereinafter "Austin").

7. ARGUMENT

A. First Ground: Claims 1-3, 5-26 and 28-45 rejected under 35 U.S.C. 103(a)

1. General

Under Section 103(a), a patent may not be obtained if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which the subject matter pertains. To establish a prima facie case of obviousness under this section, the Examiner must establish: (1) that there is some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings, (2) that there is a reasonable expectation of success, and (3) that the prior art reference (or references when combined) must teach or suggest all the claim limitations. (See e.g., MPEP 2142). As discussed in detail below, the reference(s) cited by the Examiner fail to meet these conditions.

2. Claims 1, 5-17, 19-24 and 28-45

The Examiner's rejection of Appellant's independent claim 1 as follows is representative of the Examiner's rejection of Appellant's claims as unpatentable over Alsup and Darcy:

As to claim 1, Alsup teaches a method for extended memory (the memory is extended by having more than one cache, see abstract, lines 5 - 7) support in a database system (see figure 3, data B1, data B2, and data B3) having a primary cache for storing a database page (see L1 cache, paragraph 5, line 2); when the primary cache is full, replacing database pages from the primary cache using the secondary cache (see paragraph 6, if a miss occurs in the L1... lines 5-8, and paragraph 44, lines 2-6); in response to a request for a particular database page, searching for the database particular page in the secondary cache if the particular database page is not found in the primary cache (see paragraph 6, lines 3-6), and swapping the particular database page found in the secondary cache with a database page in the primary cache, so as to replace a database page in the primary cache with the particular database page from the secondary cache (see paragraph 6, lines 11-15). The Alsup reference does not teach: using a memory mapped file, creating a secondary cache in system memory available to the database system; mapping a virtual address range to at least a portion of the secondary cache if the particular database page is found in the secondary cache, determining a virtual address in the secondary cache where the particular database page resides based on the mapping. The Darcy reference teaches using a memory mapped file, creating a secondary cache in stem memory available to the database system (see column 5, lines 9-16); mapping a virtual address range to at least a portion of the secondary cache if the particular database page is found in the secondary cache (see column 5, lines 22-32), determining a virtual address in the secondary cache where the particular database page resides based on the mapping (see column 5, lines 17-32). Therefore, it would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the Alsup system by using the Darcy system reference to create a software cache, because the implementation of software cache is not limited to one aspect, but it can be used in numerous of others applications, for sharing a volume of storage in a distributed manner. The software cache also, is programmed to operate like a set associative hardware cache to allow efficient access to block of data in the cache (see column 28, lines 60-67, and column 29, lines 1-10).

(Final Rejection, pages 2-3)

The Examiner's Final Rejection equates Alsup's microprocessor (hardware) L2 cache to Appellant's software-implemented cache in system memory that is specifically designed for improving performance of database systems. Although both Alsup's L2 cache and Appellant's claimed invention generally involve the subject of secondary caching mechanisms, there are a number of differences between Alsup's on-board microprocessor cache and Appellant's software-implemented cache in system memory that is designed for improving performance of database systems.

Alsup describes an L2 (secondary) cache mechanism that is implemented on board a microprocessor (i.e., physically located on the microprocessor chip) (Alsup, paragraph [0020] and Fig. 1 depicting L2 Cache 130 on microprocessor 100). Alsup's L2 cache is a dedicated cache that directly supports the operations of the primary (L1 or Level 1) cache, which also resides directly on the microprocessor (Alsup, paragraph [0028], paragraph [0031], Fig. 1 at 101A, 101B, and Fig. 2 at 101). As such, both the L1 and L2 caches are very specific features of microprocessors, and of course require direct support by the microprocessor. Not only is the L2 cache physically placed onto the microprocessor chip, the microprocessor includes specific firmware (i.e., hard coded instructions) to support operation of the L2 cache (see e.g., Alsup paragraphs [0020]-[0021], paragraphs [0026]-[0027], Fig. 1 at 124, 126, 140; see also, Fig. 2 at 215A, 215B, 250, 255). Clearly, a microprocessor-based L2 cache is a very specific hardware cache solution for supporting operations of the microprocessor: working in conjunction with the microprocessor L1 cache, the L2 cache improves the loading of executable code instructions so that the microprocessor may execute faster.

Appellant's claimed invention is not directed to any caching mechanism for use on a processor. Instead, Appellant's claimed invention is a software-based solution which uses system memory located external to the microprocessor. (In contrast, see, e.g., Alsup Fig. 1 showing system memory external to the microprocessor 100 and its L2 cache 130). Moreover, Appellant's solution is used to cache data pages (Appellant's specification, paragraph [0020]) -- objects that are fundamentally different than the bytes of code instruction transferred from L2 cache to L1 cache (and then to the microprocessor's execution unit) in Alsup's system. More particularly, Appellant's claimed invention uses a memory mapped file (i.e., an operating system-supported feature) to create a secondary cache in system memory, for improving the caching of database pages (i.e., specific runtime database objects) (Appellant's specification paragraph [0020], paragraph [0067]).

By way of background, a database server typically employs a cache in system memory for caching database objects (e.g., data pages for tables, indexes, and the like) during runtime operation. It is advantageous to cache this type of data in memory so that the data does not have to be brought into memory from an external source (e.g., disk) each time an operation on the data is to be performed (Appellant's specification paragraph [0065]). For example, in response to a request to fetch database data (e.g., pursuant to an SQL query), a database server may find it advantageous to cache a particular index. The foregoing "primary" cache is a software-based cache used for caching runtime program objects (Appellant's specification paragraph [0066]). It is implemented in general-purpose system memory in response to software (i.e., media-stored instructions executable on a general purpose microprocessor) loaded from disk, during operation of the software program (e.g., relational database software), and is used to cache runtime program objects (e.g., database tables, database indexes, etc.). Note that this software-based cache, which is operating at runtime to cache program <u>data</u> objects, operates concurrently and independently of any processor-based caching mechanism. A processor-based cache, in contrast, operates on raw memory units (byte block), typically functioning to load <u>code</u> (i.e., microprocessor-executable instructions) for execution by the processor's execution unit. Importantly, there is no way to use the processor-based cache to bind frequently-used runtime data objects (e.g., database tables or indexes), so that those data objects are readily accessible to the database software. Instead, the processor-based cache improves the execution of the code that comprises the database software, but does little or nothing for runtime data objects.

The primary cache that is available to an application (e.g., a database server) is often limited, and thus Appellant's invention provides a specific improvement. To the extent that the data used by the application is not available in the primary cache (especially, due to a cache size constrained to less than 4GB on 32-bit machines), it needs to be brought in from another source, namely from disk or other external storage (Appellant's specification, paragraphs [0065]). However, disk reads/writes are expensive and adversely impact application performance. Appellant's invention solves this problem by providing extended memory support though the creation of a secondary cache that acts like a swap space for the primary cache (Appellant's specification, paragraphs [0066]-[0067]).

A microprocessor cache, such as described by Alsup, does not address the above identified problem that is solved by Appellant's invention. Specifically, in certain architectural scenarios, such as Linux running on a 32-bit Intel microprocessor, it is desirable to support a very large address space (e.g., 64GB) even though the platform itself only supports a 4GB memory space (Appellant's specification, paragraph [0067]). Appellant's invention provides an approach for mapping to extended memory on various devices so that an application can bring data in from various sources without needing to know or be concerned about the underlying location of this data (Appellant's specification, paragraph [0089]). In the particular instance of Appellant's claimed invention, a secondary cache is created in system memory using a memory mapped file (Appellant's specification, paragraph [0067]). These features, for which there is no analogue found in processor-based caches, are especially important for memory intensive applications, such as large database systems, that benefit from the ability to address more memory than can be supported in the address space of a given platform (e.g., a 32-bit platform).

The use of a memory mapped file space to create a secondary cache from which the system replaces the pages from the primary cache is advantageous in that it enables the database system to avoid disk reads. If a particular page is not found in the primary cache, the secondary cache is searched to determine if it contains the page (Appellant's specification, paragraph [0105]; Fig. 5). If there is a secondary cache "hit" (i.e., if the page is cached in the secondary cache), Appellant's solution maps a virtual address window to that portion of the memory mapped secondary cache where the page resides and copies it to the primary cache (Appellant's specification, paragraph [0067]). This enables the database system avoid expensive I/O operations while making effective use of memory mapping and shared memory features of the operating system.

Appellant's claims include the above-described features that distinguish Appellant's invention from a processor-based cache mechanism such as the one described by Alsup. For example, Appellant's claim 1 includes the following claim limitations:

> A method for extended memory support in a database system having a primary cache for storing database pages, the method comprising:
> using a memory mapped file, creating a secondary cache in system memory available to the database system;
> mapping a virtual address range to at least a portion of the secondary cache;
> when the primary cache is full, replacing database pages from the primary cache using the secondary cache;
> in response to a request for a particular database page, searching for the database particular page in the secondary cache if the particular database page is not found

> in the primary cache;
> if the particular database page is found in the secondary cache, determining a virtual address in the secondary cache where the particular database page resides based on the mapping; and
> swapping the particular database page found in the secondary cache with a database page in the primary cache, so as to replace a database page in the primary cache with the particular database page from the secondary cache.
>
> (Appellant's claim 1, emphasis added)

As illustrated above, Appellant's secondary cache is created in system memory using a memory mapped file features and is used for caching database pages for a database system. As such, it is clearly distinguishable from Alsup's L2 cache which is a hardware solution physically located on a microprocessor chip supporting operations of the microprocessor.

Additionally, Appellant's claimed invention specifies the secondary cache is created in system memory using a memory mapped file (i.e., an operating system-supported shared memory file system), thereby extending available system memory (e.g., from 4GB to 64GB) without any specific hardware support. This is described, for example, in Appellant's specification as follows:

> ... the present invention makes use of a shared memory file system feature ("shmfs") and a memory mapped file feature ("mmap") available on Linux platforms. A shmfs file is created and the database system maps a virtual address range (window) to the portion of the shmfs file through the mmap interface. This mechanism enables the database system to address a portion of memory mapped shmfs file whose size can be as large as 64G.
>
> (Appellant's specification, paragraph [0067]), emphasis added)

This feature of using a memory mapped file to create a secondary cache from which the database system replaces database pages from the primary cache is also specifically included in Appellant's claims including, for instance, the following limitations of Appellant's claim 1.

> using a memory mapped file, creating a secondary cache in system memory available to the database system;
>
> (Appellant's claim 1)

The Examiner acknowledges that Alsup does not teach using a memory mapped file to create a secondary cache in system memory available to the database system and therefore adds Darcy for these teachings. More particularly, at page 8 of the Final Rejection, the Examiner references the following portion of Darcy for the teachings of using a memory mapped file to create a secondary cache in system memory:

> The computer comprises a processor programmed to implement a software cache hierarchy having at least two software caches that are interrelated to form the cache hierarchy, the at least two software caches including at least a first software cache and a second software cache, wherein the first and second software caches employ different hashing techniques for mapping an address into the first and second software caches.
>
> (Darcy, col. 5, lines 9-16).

As illustrated above, Darcy simply describes a software cache hierarchy including two or more software caches and the use of different hashing techniques for mapping an address into these caches. Respectfully, these features of Darcy are not equivalent to Appellant's claim limitations of creating a secondary cache using a memory mapped file feature. Moreover, the "caches" described by Darcy include not only caches created in system memory, but Darcy also provides that caches may be implemented using "less expensive storage resources (e.g., disks)" (Darcy, col. 30, lines 6-11). This is in direct contrast with Appellant's approach which provides extended memory support in system memory in order to avoid having to read in database pages from disk storage.

Creating a secondary cache in system memory using a memory mapped file is particularly important in for large database systems as it enables the system to address more memory than can be supported in the address space of a given platform (e.g., a 32-bit platform). Moreover, Appellant's approach enables extended memory on various devices to be used transparently so that an application can bring data in from various sources without needing to know or be concerned about the underlying location of this data (Appellant's specification, paragraph [0089]). At page 9 of the Final Rejection, the Examiner references column 12, line 65 to column 3, line 13 of Darcy for the corresponding teachings. However, here Darcy is describing a distributed system in which various blocks from a storage volume are maintained at various nodes of the distributed system (Darcy, Fig. 7 at 701, 703, 705, 707, 709, 711). Although Darcy maintains meta data describing the location of the blocks, interactions among multiple nodes may be required in order to perform operations on a given block. For example, Darcy's system may store a given block at both node 703 and node 705 (Darcy col. 13, lines 44-46). When a write on the given block is to be performed (e.g., at node 705), Darcy's system must know which other nodes have the same block and then must coordinate with these other nodes to ensure that the block is invalidated (or locked) at other nodes before the write operation may be performed at node 705 (Darcy, col. 13, lines 49-66).

With Appellant's solution, in contrast, an application can simply request a desired database page (e.g., page P2) and Appellant's system handles the process of searching for the desired page and making it available, regardless of its physical location. The page may, for example, be brought in from memory of a remote (i.e., different) machine in a database cluster. Appellant's solution maintains a mapping to pages in the secondary cache (Appellant's specification, paragraphs [0089]-[0091]). In addition to providing a translation from page name to an offset, the offset in the data structure is also a mapping which enables the system to find the actual location of where a particular page is stored in the shared cache (Appellant's specification, paragraph [0092]). These features are also described in Appellant's claims, including, for example, in the above-referenced limitations of Appellant's claim 1.

At page 9 of the Final Rejection, the Examiner references Darcy at col. 29, lines 22-33 for the corresponding teachings; however, Darcy is describing managing <u>locally</u> stored blocks and the meta data associated with those blocks (Darcy col. 29, lines 3-7). Darcy divides the local cache into cache groups (Darcy col. 29, lines 8-10) and a hashing function is used to determine the cache group in which a particular block is stored (Darcy col. 29, lines 22-30). Thus, while Darcy describes how one might implement a local cache at a particular node and find a particular block in this local cache, <u>the referenced teachings of Darcy are not describing a secondary cache, but rather is describing how one might implement a primary or local cache associated with each node of a distributed system</u> (Darcy col. 28, lines 56-63). Additionally, and as previously discussed, Darcy describes using different hashing techniques for the primary cache and the secondary cache (see, e.g., Darcy col. 5, lines 2-4). All told, the referenced teachings of Darcy are not analogous to Appellant's claim limitations of a memory mapped file feature for mapping to pages in extended memory.

3. Claims 2-3 and 25-26

Further distinctions between Appellant's solution and the prior art references are also shown in Appellant's dependent claims. For example, Appellant's claims 2 and 3 include claim limitations of creating the secondary cache using a shared memory file system provided by an operating system. Here, the Examiner references Alsup and Darcy for these teachings as follows:

> As to claim 2 Alsup discloses wherein said creating step includes creating the secondary cache. The Darcy reference teaches wherein said creating step includes creating the secondary cache (see figure 1, L2, which communicates with L1 data cache 101 B and another shared memory file.
> As to claim 3, Alsup discloses wherein the shared memory file system is available as part of an operating system on a computer platform on which the database system is running (see figure 2, system memory communicates with the data base, as part of operating system.
>
> (Final Rejection, page 3)

As illustrated above, the Examiner references Fig. 1 and Fig. 2 of Alsup as including the teachings of creating the secondary cache using a shared memory file system. However, Fig. 1 of Alsup clearly shows that the L2 secondary cache (as well as the L1 primary cache) is located on the Microprocessor 100 (i.e., implemented in hardware). In addition, neither Fig. 1, Fig. 2 or any other portion of Alsup includes any reference whatsoever to a shared memory file system (whether provided as part of an operating system or otherwise). As Alsup's L2 cache is implemented in hardware (i.e., on a microchip) it obviously cannot be created using an operating system shared memory file system. Moreover, Alsup's Fig. 1 clearly shows system memory as a separate component from an L2 cache on a microprocessor. The Examiner also notes that Darcy describes exporting a volume of storage from a root host computer to one or more child host computers, so that the root host and the child nodes share access to a volume of storage (Final Rejection, page 9). However, Applicant does not agree that this is at all analogous to Appellant's claim limitations of a shared memory file system provided by the operating system. In particular, Appellant's detailed review of the portion of Darcy referenced by the Examiner (i.e., Darcy col. 4, lines 1-14) as well as the balance of the Darcy reference also finds no mention of a shared memory file system or of creating a secondary cache using an operating system's shared memory file system. Thus, Appellant’s claim limitations of a creating a secondary cache using an operating system's shared memory file system and memory mapped file features are not taught or suggested by Alsup or Darcy.

4. Claim 18

Another distinction between Appellant’s invention and the prior art references is illustrated by Appellant's claim 18, which includes claim limitations of determining database pages to be maintained in the secondary cache based, at least in part, on workload of the database system. Appellant's invention retains frequently used runtime data objects (e.g., database tables or indexes) in cache in system memory, so that those data objects are readily accessible to support the specific transactional needs required in a database system. Only certain types of database objects are stored in the secondary cache of Appellant's invention as Appellant's invention selectively determines the appropriate items to be brought into the case based on the workload of the database system (Appellant's specification, paragraph [0093], paragraph [0102]). Furthermore, the focus of Appellant's invention is to provide extended memory support for caching database pages (e.g., for tables, indexes and the like) in system memory in order to avoid having to bring this data into memory from disk in order to perform runtime operations on the data. The caches described Alsup and Darcy, in contrast, are very generic and are not designed to supporting the transactional requirements of database systems.

The Examiner again references Fig. 2 of Alsup for the corresponding teaching of determining database pages to be maintained in the secondary cache based, at least in part, on workload of the database system (Final Rejection, page 4). However, Fig. 2 of Alsup simply shows the L1 and L2 microprocessor cache arrangement and does not even make mention of a database system or its workload.

5. Conclusion

For the reasons stated above, it is respectfully submitted that neither Alsup nor Darcy, either alone or in combination, teach or suggest all of the limitations Appellant's claims. Accordingly, it is believed that the claims 1-3, 5-26, and 28-45 distinguish over the cited art and that the Examiner's rejection of these claims under Section 103(a) should not be sustained.

B. Second Ground: Claims 4 and 27 rejected under 35 U.S.C. 103(a)

1. Claims 4 and 27

Claims 4 and 27 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Darcy (above) in view of Austin et al., (US No: 20030162544 A1, hereinafter "Austin"). Here, the Examiner relies on Darcy as teaching all of the limitations of claims 1-3 (discussed above), but adds Austin for the proposition that it teaches use of the use of a Linux operating system.

Appellant's claims are believed to be allowable for at least the reasons cited above (in Appellant's **First Ground** of appeal, which is incorporated by reference herein) pertaining to the deficiencies of Darcy (and Alsup) as to Appellant's invention. Austin does not cure any of these deficiencies. As described above, Appellant's claimed secondary cache is created using an operating system's shared memory file system (as well as its memory mapped file feature). These teachings of creating a software-implemented cache using specific operating system features are very different than Alsup's on-board processor cache. In addition, although Darcy discusses software-implemented caches, Darcy makes no mention of creating a cache using these shared memory file system and memory mapped file features of an operating system as provided in Appellant's specification and claims. Claim 4 (when read in conjunction with claims 1-3) does not simply describe use of the Linux operating system (as taught by Austin), but instead specifies a software-implemented secondary cache created in system memory using the Linux shared memory file system (as well as using Linux memory mapped file feature). Claim 27 includes similar limitations. These claim limitations of a software-based secondary cache which is created using the Linux operating system's shared memory file system and memory mapped file features are not taught or suggested by the combination of Alsup, Darcy and Austin.

2. Conclusion

For the reasons stated above, it is respectfully submitted that neither Darcy nor

Austin, either alone or in combination, teach or suggest all of the limitations Appellant's claims. Accordingly, it is believed that the claims 4 and 27 distinguish over the cited prior art references and that the Examiner's rejection of these claims under Section 103(a) should not be sustained.

C. Conclusion

Appellant's invention greatly improves the efficiency of caching of database objects for use in a database system. It is respectfully submitted that the present invention, as set forth in the pending claims, sets forth a patentable advance over the art.

In view of the above, it is respectfully submitted that the Examiner's rejection of Appellant's claims under 35 U.S.C. Section 103 should not be sustained. If needed, Appellant's undersigned attorney can be reached at 925 465 0361. For the fee due for this Appeal Brief, please refer to the attached Fee Transmittal Sheet. This Appeal Brief is submitted electronically in support of Appellant's Appeal.

Respectfully submitted,

Date: March 28, 2008

/G. Mack Riddle/

G. Mack Riddle; Reg. No. 55,572
Attorney of Record

925 465 0361
925 465 8143 FAX

8. CLAIMS APPENDIX

1. A method for extended memory support in a database system having a primary cache for storing database pages, the method comprising:

using a memory mapped file, creating a secondary cache in system memory available to the database system;

mapping a virtual address range to at least a portion of the secondary cache;

when the primary cache is full, replacing database pages from the primary cache using the secondary cache;

in response to a request for a particular database page, searching for the database particular page in the secondary cache if the particular database page is not found in the primary cache;

if the particular database page is found in the secondary cache, determining a virtual address in the secondary cache where the particular database page resides based on the mapping; and

swapping the particular database page found in the secondary cache with a database page in the primary cache, so as to replace a database page in the primary cache with the particular database page from the secondary cache.

2. The method of claim 1, wherein said creating step includes creating the secondary cache using a shared memory file system.

3. The method of claim 2, wherein the shared memory file system is available as part of an operating system on a computer platform on which the database system is running.

4. The method of claim 3, wherein the operating system comprises a Linux operating system.

5. The method of claim 1, wherein said mapping step includes using a memory mapped file function.

6. The method of claim 5, wherein the memory mapped file function is available as part of an operating system on a computer platform on which the database system is running.

7. The method of claim 1, wherein said creating step includes creating the secondary cache on external memory available to the database system.

8. The method of claim 1, wherein said swapping step includes consulting a least recently used (LRU) list maintained for the primary cache to determine the database page to be moved to the secondary cache.

9. The method of claim 8, wherein said swapping step further comprises copying the database page to be moved to the secondary cache to a temporary buffer.

10. The method of claim 9, wherein said swapping step further comprises moving the particular database page from the secondary cache to address of the database page in the primary cache to be moved to the secondary cache.

11. The method of claim 9, wherein said swapping step further comprises moving the database page from the temporary buffer to the secondary cache.

12. The method of claim 11, wherein further comprising:

adding the replaced database page to a most recently used end of a most recently used/least recently used (MRU/LRU) list maintained for the secondary cache.

13. The method of claim 1, wherein said replacing step includes maintaining a least recently used (LRU) list for the primary cache and selecting the database page to be moved to the secondary cache based on said LRU list.

14. The method of claim 1, further comprising:

providing a washing mechanism in the secondary cache for writing database pages in the secondary cache to disk.

15. The method of claim 14, wherein a database page is written from the secondary cache to disk in response to copying a database page from disk to the primary cache.

16. The method of claim 15, wherein the database page written from the secondary cache to disk is selected, based at least in part, on a most recently used/least recently used (MRU/LRU) list maintained for the secondary cache.

17. The method of claim 1, wherein said replacing step includes determining database pages to be maintained in the secondary cache.

18. The method of claim 17, wherein said determining step includes determining database pages to be maintained in the secondary cache based, at least in part, on workload of the database system.

19. The method of claim 1, wherein said replacing step includes substeps of:

moving a database page from the primary cache to the secondary cache; and

reading a database page into the primary cache from disk.

20. The method of claim 19, wherein said substep of moving a database page from the primary cache includes selecting a database page from the primary cache based on a least recently used (LRU) list maintained for the primary cache.

21. The method of claim 19, wherein said substep of moving a database page from the primary cache includes selecting a location for the database page in the secondary cache based on a most recently used/least recently used (MRU/LRU) list maintained for the secondary cache.

22. The method of claim 1, further comprising:

storing on a computer-readable medium processor-executable instructions for performing the method of claim 1.

23. The method of claim 1, further comprising:

downloading a set of processor-executable instructions for performing the method of claim 1.

24. A database system providing extended memory support, the system comprising:

a primary cache for maintaining data pages used by the database system in addressable memory available to the database system;

a secondary cache, created in system memory using a memory mapped file, for maintaining data pages replaced from the primary cache in extended memory available to the database system;

a search module for receiving a request from a user for a particular data page and determining whether the particular data page is in secondary cache if the particular data page is not in the primary cache; and

a module for replacing a data page in the primary cache with the particular data page from the secondary cache if the particular data page is found in the secondary cache.

25. The system of claim 24, wherein the secondary cache is implemented using a shared memory file system.

26. The system of claim 25, wherein the shared memory file system is available as part of an operating system on a computer platform on which the database system is running.

27. The system of claim 26, wherein the operating system comprises a Linux operating system.

28. The system of claim 24, wherein the secondary cache is mapped to the extended memory using a memory mapped file function.

29. The system of claim 28, wherein the memory mapped file function is available as part of an operating system on a computer platform on which the database system is running.

30. The system of claim 24, wherein the secondary cache is created on external memory available to the database system.

31. The system of claim 24, wherein the primary cache includes a least recently used (LRU) list for determining data pages to be moved to the secondary cache.

32. The system of claim 31, wherein the module for replacing consults the LRU list for selecting the data page to be moved to the secondary cache.

33. The system of claim 24, wherein the module for replacing copies the data page to be moved to the secondary cache to a temporary buffer.

34. The system of claim 33, wherein the module for replacing moves the particular data page from the secondary cache to address of the data page in the primary cache to be moved to the secondary cache.

35. The system of claim 33, wherein the module for replacing moves the data page from the temporary buffer to the secondary cache.

36. The system of claim 35, wherein the secondary cache includes a most recently used/least recently used (MRU/LRU) list and the module for replacing adds the data page moved to the secondary cache to the most recently used end of said MRU/LRU list.

37. The system of claim 24, further comprising:

a washing mechanism in the secondary cache for writing data pages in the secondary cache to disk.

38. The system of claim 37, wherein the washing mechanism writes a data page in the secondary cache to disk in response to copying a data page from disk to the primary cache.

39. The system of claim 38, wherein the washing mechanism selects the data page from the secondary cache based, at least in part, on a most recently used/least recently used (MRU/LRU) list maintained for the secondary cache.

40. The system of claim 24, wherein the module for replacing determines data pages to be maintained in the secondary cache.

41. The system of claim 24, further comprising:

a module for reading a page into the primary cache from disk.

42. The system of claim 41, wherein the module for reading selects a data page from the primary cache to be moved to the secondary cache if the primary cache is full.

43. The system of claim 42, wherein the module for reading selects the data page based on a least recently used (LRU) list maintained for the primary cache.

44. The system of claim 42, wherein the module for reading selects a location in the secondary cache for the data page to be moved from the primary cache.

45. The system of claim 44, wherein the module for reading selects the location in the secondary cache based on a most recently used/least recently used (MRU/LRU) list maintained for the secondary cache.

9. EVIDENCE APPENDIX

This Appeal Brief is not accompanied by an evidence submission under §§ 1.130, 1.131, or 1.132.

10. RELATED PROCEEDINGS APPENDIX

Pursuant to Appellant's statement under Section 2, this Appeal Brief is not accompanied by any copies of decisions.